UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/723,967	11/26/2003	Joseph G. Laura	IDF 2584 (4000-16100)	9521

28003 7590 09/21/2009

SPRINT
6391 SPRINT PARKWAY
KSOPHT0101-Z2100
OVERLAND PARK, KS 66251-2100

EXAMINER	
WANG, BEN C	

ART UNIT	PAPER NUMBER
2192	

MAIL DATE	DELIVERY MODE
09/21/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Continuation of 11 from the form PTO-303. does not place the application in condition for allowance because:

Response to Arguments

Applicant's arguments filed on May 18, 2009 have been fully considered but they are not persuasive.

In the remarks, Applicant argues that, for examples:

(A.1) *Nace* does not disclose a system for non-intrusively monitor variables during operation of an application (stated in bullet I section on pages 15-18)

(A.2) *Nace* does not disclose obtaining a value for one or more of the plurality of variables written to the address space by the application during the real-time operation of the application (stated in bullet II section on pages 18-20)

(A.3) *Nace* does not disclose "attaching to an address space" (stated in bullet III section on pages 20-21)

(A.4) *Nace* does not disclose using the offset to obtain a value for one or more of the plurality of variables written to the address space by the application during the real-time operation of the application (stated in bullet IV section on pages 21-22)

(A.5) *Nace* does not disclose COBOL (stated in bullet V section on pages 23-18)

(A.6) *Nace* does not disclose a technical layer (stated in bullet VI section on pages 24-25)

Examiner's response:

(R.1) As per the argument I (A.1) above, *Nace* clearly discloses a system for non-intrusively monitor variables during operation of an application (e.g., Abstract - … enables individual process to request and exchange data in a shared memory space … Processes, such as applications, or other software running under an operating system … register to blocks of a shared memory space via an administrative memory space which tracks pointers, handles and other indicators of memory areas populated by individual processes. When one process requests access to a variable, pointer or other data generated by another process … may locate the target data belonging to the other process in the shared memory space, via a lookup of relative addressing in a separate administrative memory space).

Further, *Nace* teaches an inter-process communications platform (*a technical layer – socket APIs*) enables individual processes to request and exchange data (*monitoring data*) in a shared memory space (*a technical layer – shared memory APIs*); when one process (*monitoring process*) requests access to a variable (monitored variable) by another process (*monitored process*) (e.g., abstract)

Furthermore, *Nace* teaches the requesting process (*monitoring process*) may communicate with and operate on its corresponding memory block in set of memory blocks to read such as variables (*non-intrusively monitoring*), array, table or other content (e.g., [0022])

Thus, *Nice's* reference is reasonably interpreted to meet the claim limitations of *non-intrusively monitoring variables* as explained above.

(R.2) As per the argument II (A.2) above, *Nace* teaches "an inter-process communications platform enables individual processes to request and exchange data in a shared memory space … Processes, such as applications or other software running under an operating system (*during the real-time operation of the application*) … register to blocks of a shared memory space …" (e.g., stated in Abstract, Lines 1-11); and "when one process request **access to a variable,** pointer or other data **generated by another process** (*variables written to the address space by the application*), the request is mediated by the communications engine (e.g., stated in Abstract, Lines 9-12)

Further, Nace discloses "… the requesting process … may communicate with and operate on its corresponding memory block in the set of memory blocks … to read … such as variables, arrays, tables or other content" (*obtaining a value for one or more of the plurality of variables*) (e.g., [0023]); "… a shared memory space 104 may be access by a global set of processes 102a, 102b … containing an arbitrary number of software applications …" (stated in paragraph [0015])

Thus, *Nice's* reference is reasonably interpreted to meet the claim limitations of *obtaining a value for variables written to the address space* as explained above.

(R.3) As per the argument III (A.3) above, *Nace* discloses "… a unique memory manager from the set of memory managers … may be assigned to each process in the set of processes 102a, 102b, ... and/or to each memory block in the set of memory blocks 114a, 114b ..." (stated in paragraph [0019])

Further, *Nace* teaches "… The communications engine, memory management objects and other resources may then lock the portion of the shared memory space allocated to the target process to permit the requesting process to access the data …" (*attaching to an address space*) (e.g., Abstract, Lines 11-14)

Further, *Nace* teaches "The set of memory managers … may provide an interface to the shared memory various mechanisms, which may include for instance presenting APIs or dynamic link library (dll) or other access points … Using those APIs, dlls, other interface … may generate a call to initiate a session with another process …" (e.g., [0027])

Thus, *Nace's* reference is reasonably interpreted to meet the claim limitations of *attaching to an address space* as explained above.

(R.4) As per the argument IV (A.4) above, *Nace* discloses "… the newly 0generated shared memory block in the set of memory blocks … which data may

include … variables, **address offset** or other memory mapping data and other information related to the requesting process ... **to read** … **such as variable,** arrays, tables …" (stated in paragraph [0022]); and "… **by computation of the offset** or other know relation to its associated buddy block or segment …" (stated paragraph [0039])

Further, *Nace* teaches "… a fixed memory map may be configured to be shared by a set of process with variables … being mapped into that fixed (absolute) space. Individual processes may then identify and operate on variables of interest …" (e.g., Fig. 1, element of Memory Manager – Mapping To Fixed (Absolute) Address; [0006])

Thus, *Nace's* reference is reasonably interpreted to meet the claim limitations of *using the offset to obtain a value* as explained above.

(R.5) As per the argument V (A.5) above, in response to applicant's argument of *COBOL embodiment*, a recitation of the intended use of the claimed invention must result in a structural difference between the claimed invention and the prior art in order to patentably distinguish the claimed invention from the prior art. If the prior art structure is capable of performing the intended use, then it meets the claim.

Further, the structure of *Nace* is capable of performing the intended use as *Nace* discloses "An example of computer code generating a call to initiate a session is illustrated in Fig. 3. As shown in that figure, a message-based session may be initiated by a process in the set of processes … or other entity via

communications engine … Although specific languages, routines or other code characteristics are illustrated, it will be appreciated that **different types of code or instruction may be used** …" (shown in Fig. 3; stated in paragraph [0028])

Thus, *Nace* prior art reference is capable of performing the intended use of recited COBOL embodiment, then it meets the claim.

(R.6) As per the argument VI (A.6) above, *Nace* clearly discloses "The invention may provide application programming interfaces (APIs) to create another level of data integrity since this may keep the communication engine or other process from writing directly to a shared memory block. An API may also allow processes to exchange data …" (e.g., Fig. 2, elements 120a, 120b – API to memory managers; paragraph [0008], Lines 35-42); "… a new or newly registered memory block within the set of memory blocks … may be secured or generated …" (i.e., *a shared memory routine*) (stated in Fig. 4, element 412 – Create Memory Block; in paragraph [0031]) and "The architecture as shown may also include a communications engine 108 **communicating with each process** … and **corresponding memory block** ..." (i.e., *a socket routine*) (stated in paragraph [0017])

Further, in response to applicant's argument that the references fail to show certain features of applicant's invention, it is noted that the features upon which applicant relies (i.e., technical layer APIs (including socket and shared memory) features; **NOTE:** also pointed out by Applicant on page 24, VI bullet section in REMARKS) are not recited in the rejected claim(s). Although the

claims are interpreted in light of the specification, limitations from the specification are not read into the claims. See *In re Van Geuns*, 988 F.2d 1181, 26 USPQ2d 1057 (Fed. Cir. 1993)

Thus, *Nace's* reference is reasonably interpreted to meet the claim embodiment of *the technical layer* as explained above.

/Ben C Wang/
Ben C. Wang
Examiner, Art Unit 2192